In the Claims:

Please amend claims 2, 5, 22, 25, 32, 39, and 46. The claims are as follows.

1. (Original) A method of inserting one or more global breakpoints for debugging computer software, said method including the steps of:

inserting a global breakpoint in a page containing software code if said page is present in memory;

reading said page into memory if not present in memory, and inserting a global breakpoint in said page immediately after being read into memory; and

detecting a private copy of said page if present, and inserting a global breakpoint in said private copy.

2. (Currently amended) The method according to claim 1, wherein said reading step includes the step of providing a readpage process for reading said page into memory and ~~being adapted to~~ insert for inserting a global breakpoint in said page immediately after being read into memory.

3. (Original) The method according to claim 2, wherein said readpage process is implemented as a kernel routine that is called when said page is loaded into memory.

4. (Original) The method according to claim 2, wherein said reading step includes the step of setting up an operation to insert said global breakpoint in said page immediately after said page is read into memory by an operating system, if said page is not already in memory.

5. (Currently amended) The method according to claim 4, wherein said readpage process is ~~adapted by~~ comprises changing a file specific readpage process to a wrapper routine that invokes an original readpage process and then performs said operation required.

6. (Original) The method according to claim 1, wherein said detecting step includes the step of swapping said copy to a swap device after inserting said global breakpoint in said copy.

7. (Original) The method according to claim 6, wherein said detecting step includes the further step of marking said copy as dirty after inserting said global breakpoint in said copy, whereby when swapping said copy to said swap device, said global breakpoint being present in said swapped copy.

8. (Original) The method according to claim 1, further including the step of identifying said global breakpoint using an identifier of a file and an offset in said file.

9. (Original) The method according to claim 8, wherein said file identifier is an inode.

10. (Original) The method according to claim 8, further including the step of determining if said page is present in memory using a lookup table based on said file identifier and said offset.

11. (Original) A computer-implemented apparatus for inserting one or more global breakpoints for debugging computer software, said apparatus including:

a central processing unit for executing said computer software;

memory for storing at least a portion of said computer software;

means for inserting a global breakpoint in a page containing software code if said page is present in memory;

means for reading said page into memory if not present in said memory, and inserting a global breakpoint in said page immediately after being read into memory; and

means for detecting a private copy of said page if present, and inserting a global breakpoint in said private copy.

12. (Original) The apparatus according to claim 11, wherein said reading means includes means for providing a readpage process for reading said page into said memory and being adapted to insert a global breakpoint in said page immediately after being read into memory.

13. (Original) The apparatus according to claim 12, wherein said readpage process is implemented as a kernel routine that is called when said page is loaded into said memory.

14. (Original) The apparatus according to claim 11, wherein said reading means includes means for setting up an operation to insert said global breakpoint in said page immediately after said page is read into memory by an operating system, if said page is not already in memory.

15. (Original) The apparatus according to claim 14, wherein said readpage process is adapted by changing a file specific readpage process to a wrapper routine that invokes an original readpage

process and then performs the operation required.

16. (Original) The apparatus according to claim 11, wherein said detecting means includes means for swapping said copy to a swap device after inserting said global breakpoint in said copy.

17. (Original) The apparatus according to claim 16, wherein said detecting means includes means for marking said copy as dirty after inserting said global breakpoint in said copy, whereby when swapping said copy to a swap device, said global breakpoint being present in said swapped copy.

18. (Original) The apparatus according to claim 11, further including means for identifying said global breakpoint using an identifier of a file and an offset in said file.

19. (Original) The apparatus according to claim 18, wherein said file identifier is an inode.

20. (Original) The apparatus according to claim 18, further including means for determining if said page is present in said memory using a lookup table based on said file identifier and said offset.

21. (Original) A computer program product having a computer readable medium having a computer program recorded therein for inserting one or more global breakpoints for debugging computer software, said computer program product including:

computer program code means for inserting a global breakpoint in a page containing

software code if said page is present in memory;

computer program code means for reading said page into memory if not present in said memory, and inserting a global breakpoint in said page immediately after being read into memory; and

computer program code means for detecting a private copy of said page if present, and inserting a global breakpoint in said private copy.

22. (Currently amended) The computer program product according to claim 21, wherein said computer program code means for reading includes computer program code means for providing a readpage process for reading said page into said memory and ~~being adapted to insert~~ for inserting a global breakpoint in said page immediately after being read into memory.

23. (Original) The computer program product according to claim 22, wherein said readpage process is implemented as a kernel routine that is called when said page is loaded into said memory.

24. (Original) The computer program product according to claim 22, wherein said computer program code means for reading includes computer program code means for setting up an operation to insert said global breakpoint in said page immediately after said page is read into memory by an operating system, if said page is not already in memory.

25. (Currently amended) The computer program product according to claim 22, wherein said

readpage process ~~is adapted by~~ comprises changing a file specific readpage process to a wrapper routine that invokes an original readpage process and then performs said operation required.

26. (Original) The computer program product according to claim 21, wherein said computer program code means for detecting includes computer program code means for swapping said copy to a swap device after inserting said global breakpoint in said copy.

27. (Original) The computer program product according to claim 26, wherein said computer program code means for detecting includes computer program code means for marking said copy as dirty after inserting said global breakpoint in said copy, whereby when swapping said copy to a swap device, said global breakpoint being present in said swapped copy.

28. (Original) The computer program product according to claim 21, further including computer program code means for identifying said global breakpoint using an identifier of a file and an offset in said file.

29. (Original) The computer program product according to claim 28, wherein said file identifier is an inode.

30. (Original) he computer program product according to claim 28, further including computer program code means for determining if said page is present in said memory using a lookup table based on said file identifier and said offset.

31. (Original) A method of removing one or more global breakpoints for debugging computer software, said method including the steps of:

removing a global breakpoint in a page containing software code if said page containing said global breakpoint is present in memory; and

detecting a private copy of said page if present, reading said page into memory if not present in memory, and removing a global breakpoint in said private copy.

32. (Currently amended) The method according to claim 31, wherein said reading step includes the step of providing a readpage process for reading said page into memory and ~~being adapted to remove~~ for removing a global breakpoint in said page immediately after being read into memory.

33. (Original) The method according to claim 32, wherein said readpage process is implemented as a kernel routine that is called when said page is loaded into memory.

34. (Original) The method according to claim 31, wherein said reading step includes the step of turning off an operation set up earlier for inserting a global breakpoint in said page when said page is read into memory.

35. (Original) The method according to claim 31, further including the step of identifying said global breakpoint using an identifier of a file and an offset in said file.

36. (Original) The method according to claim 35, wherein said file identifier is an inode.

37. (Original) The method according to claim 35, further including the step of determining if said page is present in memory using a lookup table based on said file identifier and said offset.

38. (Original) A computer-implemented apparatus for removing one or more global breakpoints for debugging computer software, said apparatus including:

a central processing unit for executing said computer software;

memory for storing at least a portion of said computer software;

means for removing a global breakpoint in a page containing software code if said page containing said global breakpoint is present in memory; and

means for detecting a private copy of said page if present, reading said page into memory if not present in said memory, and removing a global breakpoint in said private copy.

39. (Currently amended) The apparatus according to claim 38, wherein said reading means includes means for providing a readpage process for reading said page into said memory and ~~being adapted to remove~~ for removing a global breakpoint in said page immediately after being read into memory.

40. (Original) The apparatus according to claim 39, wherein said readpage process is implemented as a kernel routine that is called when said page is loaded into said memory.

41. (Original) The apparatus according to claim 38, wherein said reading means includes means for turning off an operation set up earlier for inserting a global breakpoint in said page when said

page is read into memory.

42. (Original) The apparatus according to claim 38, further including means for identifying said global breakpoint using an identifier of a file and an offset in said file.

43. (Original) The apparatus according to claim 42, wherein said file identifier is an inode.

44. (Original) The apparatus according to claim 42, further including means for determining if said page is present in said memory using a lookup table based on said file identifier and said offset.

45. (Original) A computer program product having a computer readable medium having a computer program recorded therein for removing one or more global breakpoints for debugging computer software, said computer program product including:

computer program code means for removing a global breakpoint in a page containing software code if said page containing said global breakpoint is present in memory; and

computer program code means for detecting a private copy of said page if present, reading said page into memory if not present in said memory, and removing a global breakpoint in said private copy.

46. (Currently amended) The computer program product according to claim 45, wherein said computer program code means for reading includes computer program code means for providing

a readpage process for reading said page into said memory and ~~being adapted to remove~~ for removing a global breakpoint in said page immediately after being read into memory.

47. (Original) The computer program product according to claim 45, wherein said readpage process is implemented as a kernel routine that is called when said page is loaded into said memory.

48. (Original) The computer program product according to claim 46, wherein said computer program code means for reading includes turning off an operation set up earlier for inserting a global breakpoint in said page computer program code means for when said page is read into memory.

49. (Original) The computer program product according to claim 45, further including computer program code means for identifying said global breakpoint using an identifier of a file and an offset in said file.

50. (Original) The computer program product according to claim 49, wherein said file identifier is an inode.

51. (Original) The computer program product according to claim 49, further including computer program code means for determining if said page is present in said memory using a lookup table based on said file identifier and said offset.